AUGUST 1991 ISSUE # 36

VZ 200/300

HUNTER VALLEY VZ JOURNAL

ED ADDRESS CHANGE
DEFINING VARIABLES
FAST BASIC
DOUBLE PRECISION
144K-528K EXPANSION

PRODUCED BI-MONTHLY BY H.V.VZ.U.G.
A NON PROFIT ORGANIZATION

** EDITOR'S COMMENTS **

APOLOGIES:
VZ LOTTO DISK MODS HELD OVER TILL NEXT ISSUE DUE TO LACK OF TIME, ED.

DISCLAIMER: EVERY EFFORT IS MADE TO INSURE THE ACCURACY OF INFORMATION CONTAINED WITHIN BE IT GENERAL, TECHNICAL, PROGRAMMING, ETC. NO RESPONSIBILITY CAN BE ACCEPTED BY HUNTER VALLEY VZ USERS' GROUP OR AUTHOR AS A RESULT OF APPLYING SUCH INFORMATION IN PRACTICE.

EDITOR ADDRESS CHANGE:

As some of you may be aware my house has been sold and I'll be moving out in a few weeks time. Till I get settled in my new place please send all mail meant for me or the club to:

J.P. LEON or H.V.VZ.U.G.
PO BOX 161 JESMOND 2299

I'll advise my new address and phone number in a future issue. In the meantime try the old number or write to the PO Box number.

Getting ready to move has been very time consuming and tiring and as a result I'm way behind in answering mail and building planned projects. It will take me a while to catch up, so please be patient and I'll get back to you as soon as I can, Ed.

WANTED TO BUY/EXCHANGE:

Wanted to buy VZ 300 in good working order or will trade VZ 200 for VZ 300. If you can help please contact:

STEPHEN GAYST
12/43 JOHNSON STREET CHATSWOOD 2067
PHONE: (02) 411 6791

DISK PROBLEM 1 BY BEN HOBSON:

I bought a pack of DSE RED SPOT disks and the write protect tabs were the aluminium type. When put on a disk, it wouldn't fit in the drive. **THE ULTIMATE WRITE PROTECT.**

SOLUTION: If you have similiar problems with your disks then try thinner write protect tabs.

DISK PROBLEM 2:

Ross Woods had problems with NASHUA disks which would give READ/WRITE problems when used with the VZ most of the time. No problems with flip side of disk though.

COMPUTEC DISKS FROM KMART:

I bought several boxes of these disks and most were concave even though they came in sturdy plasic cases. Needless to say I found them to be unreliable for use with the VZ.

SOLUTION: Try different brand of disk if the brand you're using is constantly giving you READ/WRITE problems.

FINAL VZ CLEARANCE SALE:

Dick Smith is having a VZ clearance sale at its Penrith store which started about mid July. Not much in the way of hardware I believe, mainly software. I haven't had a chance to go down as yet for a look see.

For more details contact:
Dick Smith - Penrith Store - Phone: (047) 323 400

Some stores may still hold some stocks of various VZ items but they'll be few and far between. VZ 300's, Disk Drives and Controllers are about non existent as far as I know. The time is nearly here when the only support VZ users will have is from VZ user groups and publications and how long they continue to exist is up to you.

DEFINING INT-STR-SNG-DBL VARIABLES

THERE ARE THREE VARIABLE TYPES IN COMMON USE WHICH ALL OF YOU SHOULD BE FAMILIAR WITH, WHICH ARE:

```
 A-Z  - SINGLE PRECISION VARIABLES.
A$-Z$ - STRING VARIABLES.
A%-Z% - INTEGER VARIABLES.
```

THIS GIVES US 78 VARIABLES, 26 IN EACH RANGE, BUT CAN BE INCREASED TO MANY HUNDREDS BY USING TWO CHARACTER VARIABLES, EG:

```
A0-A9 - AA-AZ - A0$-A9$ - AA$-AZ$ - A0%-A9% - AA%-AZ%
B0-B9 - BA-BZ - B0$-B9$ - BA$-BZ$ - B0%-B9% - BA%-BZ%
C0.... Y0.... CA.... YA.... ETC....
Z0-Z9 - ZA-ZZ - Z0$-Z9$ - ZA$-ZZ$ - Z0%-Z9% - ZA%-ZZ%
```

NOTE: A NUMBER CANNOT BE USED AS A VARIABLE BUT CAN BE USED AS SECOND CHARACTER IN A VARIABLE AS THE EXAMPLES ABOVE SHOW.

THAT GIVES US ANOTHER 2808 VARIABLES FOR A TOTAL OF 2886, MANY MORE THAN YOU'LL EVER USE. THAT BY NO MEANS IS ALL THE VARIABLES AVAILBLE TO THE VZ USER. ALL THE SINGLE PRECISION VARIABLES STARTING WITH A-Z CAN BE DEFINED (CHANGED) TO ONE OF FOUR VARIABLE TYPES. SEE POKING TO BETTER BASIC ARTICLE FOR AN EXPLANATION.

I'VE SEEN THREE OR MORE CHARACTERS USED FOR A VARIABLE NAME, BUT I STRONGLY ADVISE AGAINST ITS USE AS IT CAN GET YOU INTO TROUBLE, EG:

```
10 ABC=12345:ABD=23456:ABZ=34567:PRINT ABC;ABD;ABZ
34567  34567  34567
```

ONLY THE FIRST TWO CHARACTERS IN A VARIABLE NAME ARE VALID AND IN ABOVE EXAMPLE THE VZ ONLY RECOGNISES 'AB' WHICH ALL THREE VARIABLES SHARE, SO ABC & ABD = ABZ BECAUSE ABZ WAS THE LAST ONE DEFINED.

NOTE: THE VZ RECOGNIZES VARIABLES A, A$ & A% THROUGH TO Z, Z$ & Z% AND ALL TWO CHARACTER VARIABLES STARTING WITH THE SAME FIRST CHARACTER AS ALL BEING DIFFERENT VARIABLES. THERE ARE TWO EXEPTIONS:

1)
```
10 DEFINT A
20 A=123:A%=999:PRINT A;A%
999 999
```

BOTH 'A' & 'A%' HAVE BEEN DEFINED AS AN INTEGER VARIABLE WHICH MEANS BOTH VARIABLES ARE SAME. SO A = A% AND 999 GETS PRINTED.

DO NOT MIX DEFINT A-Z AND A%-Z% IN THE SAME PROGRAM.

2)
```
10 DEFSTR B
20 B="TEST 1 ":B$="TEST 2 ":PRINT B;B$
TEST 2  TEST 2
```

BOTH 'B' & 'B$' HAVE BEEN DEFINED AS A STRING VARIABLE WHICH MEANS BOTH VARIABLES ARE THE SAME. SO B = B$ AND TEST 2 GETS PRINTED.

DO NOT MIX DEFSTR A-Z AND A$-Z$ IN THE SAME PROGRAM.

NOTE: WHEN TWO VARIABLES ARE THE SAME, THE LAST ONE DECLARED AS IN ABOVE EXAMPLES HOLDS THE VARIABLE CONTENT BE IT NUMERIC OR TEXT.

POKING TO BETTER BASIC

BY JOE LEON

AS MOST OF YOU MAY BE AWARE THE VZ HAS A HOST OF HIDDEN FUNCTIONS WHICH CAN BE ASSESED OR ACTIVATED BY HAVING AN EXTENDED BASIC PROGRAM OR EPROM INSTALLED. FOR THOSE THAT HAVEN'T EITHER A POKE OR TWO IN SOME CASES WILL DO THE SAME. THE POKE FUNCTION ALLOWS VZ USERS TO POKE (WRITE) TO A PARTICULAR MEMORY LOCATION WHICH MUST BE IN RAM.

IN THE COMMUNICATION REGION OF RAM THERE ARE 26 ADDRESSES, ONE ADDRESS FOR EACH VARIABLE FROM A TO Z WHICH CAN BE DEFINED (CHANGED) IN TWO WAYS FROM BASIC. ONE IS USING THE EXTENDED BASIC FUNCTIONS **DEFINT, DEFSTR, DEFSNG and DEFDBL.**

THE OTHER IS USING POKE FUNCTION TO POKE ONE OF FOUR VARIABLE TYPE VALUES. BELOW IS A LIST OF THE FOUR DEFINE FUNCTIONS, THEIR VALUES AND THE VARIABLES A TO Z AND THEIR PEEK/POKE ADDRESSES.

DEFINT=2 DEFSTR=3 DEFSNG=4 DEFDBL=8

VAR - POKE ADDRESS	VAR - POKE ADDRESS
A = POKE 30977	N = POKE 30990
B = POKE 30978	O = POKE 30991
C = POKE 30979	P = POKE 30992
D = POKE 30980	Q = POKE 30993
E = POKE 30981	R = POKE 30994
F = POKE 30982	S = POKE 30995
G = POKE 30983	T = POKE 30996
H = POKE 30984	U = POKE 30997
I = POKE 30985	V = POKE 30998
J = POKE 30986	W = POKE 30999
K = POKE 30987	X = POKE 31000
L = POKE 30988	Y = POKE 31001
M = POKE 30989	Z = POKE 31002

TYPE IN THE LITTLE PROGRAM BELOW AN RUN IT. YOU'LL SEE ADDRESSES 30977 TO 31002 DISPLAYED ON THE SCREEN TOGETHER WITH 4'S ALONG SIDE EACH ADDRESS. WHAT THAT MEANS IS THAT ALL VARIABLES FROM A TO Z ARE DEFINED AS DEFSNG (SINGLE PRECISION) TYPE AND IS THE DEFAULT VALUE.

```
10 CLS:FOR X = 30977 TO 31002
20 PRINT" ";X;PEEK(X),:NEXT X
```

ANYTIME YOU RUN A PROGRAM OR USE THE CLEAR FUNCTION ALL VARIABLES ARE RESET TO THEIR DEFAULT VALUES. WHAT THAT MEANS IS THAT YOU HAVE TO DEFINE THE VARIABLE TYPE SOMEWHERE IN THE PROGRAM YOURSELF.

DEFINING INTEGER VARIABLES:

```
10 DEFINT A or 10 POKE 30977,2
20 REM DEFINES AS INTEGER ANY VARIABLE STARTING WITH 'A'
30 A=12345.123:AB=12345.987:PRINT A;AB
12345    12345
```

NOTE: INTEGER = ANY WHOLE NUMBER WITH NO FRACTIONS OR DECIMAL PLACES IN THE RANGE OF -32768 TO 32767.
INTEGER ONLY HANDLES WHOLE NUMBERS WITH NO ROUNDING OFF.
A%-Z% CAN BE USED ALSO, BUT AS MENTIONED BEFORE,
DO NOT USE DEFINT A-Z AND A%-Z% IN THE SAME PROGRAM.

DEFINING STRING VARIABLES:

```
    10 DEFSTR B OR 10 POKE 30978,3
    20 REM DEFINES AS STRING ANY VARIABLE STARTING WITH 'B'.
    30 B="HELLO":BC="12345":PRINT B;BC
    HELLO  12345
```

NOTE: STRING = ANY TEXT, NUMERIC OR GRAPHIC CHARACTER/S
WHICH ARE ENCLOSED WITHIN QUOTES, EG: ("TEXT 1).
NO '$' SIGN AFTER B OR BC.
A$-Z$ CAN BE USED ALSO, BUT AS MENTIONED BEFORE,
DO NOT USE DEFSTR A-Z AND A$-Z$ IN THE SAME PROGRAM.

DEFINING SINGLE PRECISION VARIABLES:

```
    10 DEFSNG C OR 10 POKE 30979,4
    20 REM DEFINES AS SINGLE PRECISION ANY VARIABLE STARTING WITH 'C'
    30 C=1.23456:C0=9.87654:CA!=4.567:PRINT C;C0;CA!
    1.23456  9.87654  4.567
```

NOTE: SINGLE PRECISION = 6 DIGIT NUMERIC ACCURACY.
UP TO 6 DECIMAL PLACES ARE SHOWN.

DEFINING DOUBLE PRECISION VARIABLES:

```
    10 DEFDBL D OR 10 POKE 30980,8
    20 REM DEFINES AS DOUBLE PRECISION ANY VARIABLE STARTING WITH 'D'
    30 C=1/3:D=1/3:PRINT C:PRINT D
    1.333333
    1.3333333432674408
```

NOTE: DOUBLE PRECISION = 16 DIGIT NUMERIC ACCURACY.
FOR MORE ACCURACY ON DOUBLE PRECISION MATHS SEE
ARTICLE ON DOUBLE PRECISION ARITHMETIC.

DEFINING MORE VARIABLES:

SO FAR I'VE SHOWN HOW TO DEFINE ONLY A SINGLE VARIABLE AT A TIME. IT'S FAIRLY EASY TO DEFINE A WHOLE RANGE AT A TIME, EG:

```
    10 DEFINT A-M,Q
    20 FOR Z=30977 TO 30989:POKE Z,2:NEXT:POKE 30993,2
```

THE FIRST EXAMPLE IS EASY, BUT IN THE SECOND YOU HAVE TO KNOW THE ADDRESSES OF THE VARIABLES YOU WISH TO DEFINE. IT'S MORE TEDIOUS, BUT ACHIEVES EXACTLY THE SAME RESULT FOR THOSE WITHOUT OUT AN EXTENDED BASIC PROGRAM OR EPROM.

IN PRACTICE YOU WOULD USE THE '$' AND '%' TO DEFINE YOUR VARIABLES AS THEY ARE EASY TO SEE AND UNDERSTAND. DOUBLE PRECISION HAS TO BE DEFINED EITHER BY A POKE OR THE EXTENDED BASIC COMMAND DEFDBL.

IN CASE OF EXISTING PROGRAMS WHERE YOU'LD LIKE TO DEFINE AS INTEGER ALL SINGLE PRECISION VARIABLES YOU WOULD USE ONE OF THE TWO EXAMPLES ABOVE RATHER THAN TO PHYSICALLY ADD THE '%' SIGN TO EACH.

FORMATT SOURCE CODE

BY DAVE MITCHELL

```
00001 ;FORMATT TRK #
00002       LD    HL,ST
00003       LD    (788EH),HL
00004 ST    DI
00005       CALL  3450H
00006       EI
00007       LD    A,(IY+20)
00008       OR    A
00009       JR    Z,IPUT
00010       DI
00011       PUSH  AF
00012       CALL  4008H
00013       POP   AF
00014       LD    B,A
00015       CALL  403EH
00016       CALL  400BH
00017 IPUT  LD    HL,MES1
00018       CALL  2B75H
00019       CALL  053AH
00020       JP    C,1DBEH
00021       LD    A,(DE)
00022       CP    20H
00023       JP    Z,1A19H
00024       INC   HL
00025 ATB   LD    B,2
00026       LD    DE,0
00027 ATB1  LD    A,(HL)
00028       CP    30H
00029       JR    C,ATB4
00030       CP    3AH
00031       JR    C,ATB2
00032       CP    41H
00033       JR    C,ATB4
00034       CP    47H
00035       JR    NC,ATB4
00036       SUB   7
00037 ATB2  AND   0FH
00038       LD    C,B
00039       LD    B,4
00040 ATB3  SLA   E
00041       RL    D
00042       DJNZ  ATB3
00043       OR    E
00044       LD    E,A
00045       LD    B,C
00046       RST   10H
00047       DJNZ  ATB1
00048 ATB4  LD    A,B
00049       CP    2
00050       JP    NC,IPUT
00051       LD    A,E
00052       CP    28H
00053       JR    NC,ST
00054       DI
00055       PUSH  AF
00056       LD    (HL2),A
00057       LD    (IY+18),A
00058       LD    HL,IX1
00059       ADD   A,(HL)
00060       LD    (HL4),A
00061       LD    A,(HL)
00062       LD    (HL3),A
00063       LD    (IY+17),A
00064       CALL  4008H
00065       POP   AF
00066       OR    A
00067       JR    Z,A1
00068       LD    B,A
00069       CALL  403BH
00070 A1    IN    A,(13H)
00071       BIT   7,A
00072       JR    Z,A2
00073       CALL  400BH
00074       EI
00075       LD    HL,WP
00076       CALL  2B75H
00077       CALL  0049H
00078       DI
00079       CALL  3450H
00080       CALL  4008H
00081       LD    BC,0064H
00082       CALL  4038H
00083       JR    A1
00084 A2    LD    BC,03E8H
00085       CALL  4038H
00086       PUSH  IY
00087       POP   HL
00088       LD    DE,004DH
00089       ADD   HL,DE
00090       LD    (IY+0EH),L
00091       LD    (IY+0FH),H
00092       EX    DE,HL
00093       LD    HL,HL1
00094       LD    BC,0018H
00095       LDIR
00096       LD    H,D
00097       LD    L,E
00098       LD    (HL),00
00099       INC   DE
00100       LD    BC,0082H
00101       LDIR
00102       LD    (IY+38H),11H
00103       LD    BC,0190H
00104       CALL  4038H
00105       LD    L,(IY+0EH)
00106       LD    H,(IY+0FH)
00107       LD    DE,000BH
00108       ADD   HL,DE
00109       LD    D,H
00110       LD    E,L
00111       INC   DE
00112       LD    B,D
00113       LD    C,E
00114       INC   BC
00115       EXX
00116 INI3  LD    BC,0064H
00117       CALL  4038H
00118       LD    A,(IY+33H)
00119       RES   6,A
00120       LD    (IY+33H),A
```

```
00121        OUT   (10H),A
00122        LD    BC,0064H
00123        CALL  4038H
00124        LD    IX,IX1
00125 INI1   LD    L,(IY+0EH)
00126        LD    H,(IY+0FH)
00127        LD    D,(IY+33H)
00128        LD    B,9AH
00129        CALL  INI4
00130        LD    (IY+33H),D
00131        EXX
00132        LD    A,(IX+01)
00133        INC   IX
00134        LD    (DE),A
00135        ADD   A,(HL)
00136        LD    (BC),A
00137        LD    A,(DE)
00138        EXX
00139        CP    0FFH
00140        JP    NZ,INI1
00141        CALL  400BH
00142        EI
00143        JP    ST
00144 INI4   LD    C,(HL)
00145        LD    A,20H
00146        XOR   D
00147        RL    C
00148        JP    NC,INI5
00149        OUT   (10H),A
00150        XOR   20H
00151        LD    D,A
00152        DEC   HL
00153        OUT   (10H),A
00154        JP    INI6
00155 INI5   OUT   (10H),A
00156        XOR   00
00157        LD    D,A
00158        DEC   HL
00159        OUT   (10H),A
00160        JP    INI6
00161 INI6   INC   HL
00162        JP    INI7
00163 INI7   JP    INI8
00164 INI8   IN    A,(12H)
00165        LD    A,20H
00166        XOR   D
00167        RL    C
00168        JP    NC,INI9
00169        OUT   (10H),A
00170        XOR   20H
00171        LD    D,A
00172        DEC   HL
00173        OUT   (10H),A
00174        JP    IN10
00175 INI9   OUT   (10H),A
00176        XOR   00
00177        LD    D,A
00178        DEC   HL
00179        OUT   (10H),A
00180        JP    IN10
```

```
00181 IN10   INC   HL
00182        JP    IN11
00183 IN11   JP    IN12
00184 IN12   IN    A,(12H)
00185        LD    A,20H
00186        XOR   D
00187        RL    C
00188        JP    NC,IN13
00189        OUT   (10H),A
00190        XOR   20H
00191        LD    D,A
00192        DEC   HL
00193        OUT   (10H),A
00194        JP    IN14
00195 IN13   OUT   (10H),A
00196        XOR   00
00197        LD    D,A
00198        DEC   HL
00199        OUT   (10H),A
00200        JP    IN14
00201 IN14   INC   HL
00202        JP    IN15
00203 IN15   JP    IN16
00204 IN16   IN    A,(12H)
00205        LD    A,20H
00206        XOR   D
00207        RL    C
00208        JP    NC,IN17
00209        OUT   (10H),A
00210        XOR   20H
00211        LD    D,A
00212        DEC   HL
00213        OUT   (10H),A
00214        JP    IN18
00215 IN17   OUT   (10H),A
00216        XOR   00
00217        LD    D,A
00218        DEC   HL
00219        OUT   (10H),A
00220        JP    IN18
00221 IN18   INC   HL
00222        JP    IN19
00223 IN19   JP    IN20
00224 IN20   IN    A,(12H)
00225        LD    A,20H
00226        XOR   D
00227        RL    C
00228        JP    NC,IN21
00229        OUT   (10H),A
00230        XOR   20H
00231        LD    D,A
00232        DEC   HL
00233        OUT   (10H),A
00234        JP    IN22
00235 IN21   OUT   (10H),A
00236        XOR   00
00237        LD    D,A
00238        DEC   HL
00239        OUT   (10H),A
00240        JP    IN22
```

```
00241 IN22 INC  HL
00242      JP   IN23
00243 IN23 JP   IN24
00244 IN24 IN   A,(12H)
00245      LD   A,20H
00246      XOR  D
00247      RL   C
00248      JP   NC,IN25
00249      OUT  (10H),A
00250      XOR  20H
00251      LD   D,A
00252      DEC  HL
00253      OUT  (10H),A
00254      JP   IN26
00255 IN25 OUT  (10H),A
00256      XOR  00
00257      LD   D,A
00258      DEC  HL
00259      OUT  (10),A
00260      JP   IN26
00261 IN26 INC  HL
00262      JP   IN27
00263 IN27 JP   IN28
00264 IN28 IN   A,(12H)
00265      LD   A,20H
00266      XOR  D
00267      RL   C
00268      JP   NC,IN29
00269      OUT  (10H),A
00270      XOR  20H
00271      LD   D,A
00272      DEC  HL
00273      OUT  (10H),A
00274      JP   IN30
00275 IN29 OUT  (10H),A
00276      XOR  00
00277      LD   D,A
00278      DEC  HL
00279      OUT  (10H),A
00280      JP   IN30
00281 IN30 INC  HL
00282      JP   IN31
00283 IN31 JP   IN32
00284 IN32 IN   A,(12H)
00285      LD   A,20H
00286      XOR  D
00287      RL   C
00288      JP   NC,IN33
00289      OUT  (10H),A
00290      XOR  20H
00291      LD   D,A
00292      DEC  HL
00293      OUT  (10H),A
00294      JP   IN34
00295 IN33 OUT  (10H),A
00296      XOR  00
00297      LD   D,A
00298      DEC  HL
00299      OUT  (10H),A
00300      JP   IN34
```

```
00301 IN34 INC  HL
00302      INC  HL
00303      NOP
00304      DEC  B
00305      JP   NZ,INI4
00306      RET
00307 HL1  ADD  A,B
00308      ADD  A,B
00309      ADD  A,B
00310      ADD  A,B
00311      ADD  A,B
00312      ADD  A,B
00313      NOP
00314      CP   0E7H
00315      DEFB 18H
00316      DEFB 0C3H
00317 HL2  NOP
00318 HL3  NOP
00319 HL4  NOP
00320      ADD  A,B
00321      ADD  A,B
00322      ADD  A,B
00323      ADD  A,B
00324      ADD  A,B
00325      NOP
00326      JP   0E718H
00327      DEFB 0FEH
00328 IX1  DEFB 0
00329      DEFB 8
00330      DEFB 1
00331      DEFB 9
00332      DEFB 2
00333      DEFB 0AH
00334      DEFB 3
00335      DEFB 0BH
00336      DEFB 4
00337      DEFB 0CH
00338      DEFB 5
00339      DEFB 0DH
00340      DEFB 6
00341      DEFB 0EH
00342      DEFB 7
00343      DEFB 0FH
00344      DEFB 255
00345 MES1 DEFB 1FH
00346 *       SINGLE TRACK FORMATT*
00347      DEFB 0DH
00348 *          BY  D.MITCHELL*
00349      DEFB 0DH
00350 *ENTER TRACK NUMBER :*
00351      NOP
00352 WP   EQU  $
00353      DEFB 13
00354 *   REMOVE WRITE PRCTECT LA*
00355 *BEL*
00356      DEFB 13
00357 *        PRESS RETURN*
00358      DEFB 13
00359      NOP
```

FAST BASIC PART I BY BOB KITCH 36/10

Connecting Machine Code to Basic.

I HAVE WRITTEN ABOUT MY FAST BASIC TECHNIQUE AND USED IT IN MY LIVENUP AND SOUND SERIES OF PROGRAMS FOR THE VZ. IT IS A HYBRID LANGUAGE. A NUMBER OF USERS HAVE ALSO ASKED HOW IT IS DONE. I WILL ENDEAVOUR TO PROVIDE AN INTERESTING AND ILLUMINATING DISCUSSION OF THE TECHNIQUE.

THIS IS AN INVITATION TO EXPLORE THIS SIGNIFICANT ENHANCEMENT OF NORMAL BASIC. FOR PROGRAMMERS WHO ARE LOOKING FOR BETTER (SMARTER OR FASTER) WAYS TO WRITE PROGRAMS, THEN THIS IS OF INTEREST. ALSO, FOR BASIC PROGRAMMERS WHO ARE WRESTLING WITH Z80 MACHINE CODE, THEN FAST BASIC PROVIDES AN IDEAL INTRODUCTION.

TO FULLY UNDERSTAND FAST BASIC WE NEED TO EXPLORE -

1. HOW A BASIC PROGRAM IS STRUCTURED IN MEMORY,
2. HOW TO RESERVE AN AREA OF MEMORY FOR MACHINE CODE, AND
3. UNDERSTAND HOW TO CONNECT MACHINE CODE WITH A BASIC PROGRAM.

BEFORE I COMMENCE, IT IS WORTH MENTIONING SOME BOOKS, THAT MOST USERS WILL HAVE IN THEIR COLLECTIONS AND THAT WILL REINFORCE THE EXPLANATIONS THAT I PROVIDE. THE DSE TECHNICAL MANUALS FOR THE VZ 200 AND 300 (1983 AND 1985) ARE USEFUL. STEVE OLNEY'S BOOK (1987) ON ASSEMBLY LANGUAGE FOR BEGINNERS IS ALSO VERY USEFUL ON THESE TOPICS.

FOR A WELL-PACED INTRODUCTION TO Z80 ASSEMBLY LANGUAGE PROGRAMMING, THE TWO TANDY/RADIO SHACK BOOKS BY BILL BARDEN (1979 AND 1982) FOR THE TRS-80 ARE AS GOOD AS ANY, PARTICULARLY SINCE THE ROM IN THE TRS-80 AND VZ IS VERY SIMILAR. YOU MAY HAVE TO HUNT AROUND FOR COPIES OF THESE BOOKS AS THEY ARE OUT OF PRINT.

LET'S START BY LOOKING AT THE STRUCTURE OF A BASIC PROGRAM IN MEMORY.

1. FIVE BASIC TABLES & MEMORY UTILIZATION.

BY KNOWING THE ORGANIZATION OF A PROGRAM IN MEMORY, WE CAN MAKE IT PERFORM MORE EFFICIENTLY. ALTHOUGH YOU HAVE PROBABLY NEVER THOUGHT OF IT THIS WAY, (AND IT IS CERTAINLY NOT MENTIONED IN THE VZ MANUALS!) A BASIC PROGRAM RUNNING ON THE VZ ACTUALLY CONSISTS OF FIVE TABLES. BASIC PROGRAMMERS ARE VERY FAMILIAR WITH THE FIRST OF THESE, THE PROGRAM STATEMENT TABLE (PST), AS THIS IS THE ACTUAL TOKENIZED (COMPRESSED) PROGRAM, OR SOURCE, THAT IS WRITTEN.

THE OTHER TABLES ARE "WRITTEN" BY THE INTERPRETER WHEN THE PROGRAM IS EXECUTED. THE VARIABLE LIST TABLE (VLT) ACTUALLY CONSISTS OF TWO PARTS, THE SIMPLE VARIABLE TABLE (SVT) AND THE DIMENSIONED VARIABLE TABLE (DVT). THESE ARE POSITIONED ABOVE THE PST AND "GROW" AS THE PROGRAM EXECUTES AND "DISCOVERS" MORE VARIABLES - THAT IS THEY ARE DYNAMIC WHEREAS THE PST IS STATIC ONCE EXECUTION COMMENCES. (THE PST GROWS AS YOU WRITE THE PROGRAM).

THE REMAINING TWO TABLES ARE LOCATED BELOW THE TOP-OF-MEMORY (TOM). (FOR DOS SYSTEMS, THE TOM IS LOCATED BELOW THE DOS VECTOR) THE UPPERMOST ONE IS THE STRING AREA THAT STORES THE STRING VARIABLES USED BY A BASIC PROGRAM. NOTE THAT QUOTED STRINGS ARE STORED IN THE PST. THE NUMBER OF BYTES RESERVED IS FIXED BY THE CLEAR COMMAND AND THE DEFAULT IS 50 BYTES. IT IS A STATIC TABLE.

BELOW THE STRING AREA IS THE STACK WHICH "GROWS" DOWNWARDS IN MEMORY AND IS DYNAMIC OR CHANGEABLE. THE STACK IS USED BY THE Z80 FOR PUSHING AND POPPING REGISTERS AND ALSO BY THE BASIC PROGRAM TO KEEP TRACK OF GOSUB/RET AND FOR/NEXT CALLS.

This completes the five tables used by Basic. The intervening area of RAM below the Stack and above the VLT is called Free Space List (FSL) or unused memory. It is bounded above and below by dynamic tables (Stack and VLT) and the size of FSL changes as program execution proceeds.

(You may want to read the preceding paragraphs a few times, as it gives a rather "different" view of a Basic program to what you are probably used to.)

A memory map of this situation with the <pointers> indicating the start and end of the various tables would be as follows -

```
    ===========================! <IY REG>
    ! Top of Physical Memory    !
    ! DOS Vector on disk systems!
  B ===========================! <78B1/2H>
  A ! String Area               !
  S !---------------------------! <78A0/1H>
  I ! Stack                     !
  C !---------------------------! <78E8/9H>
    !                           !
  W ! Free Space List           !
  O !                           !
  R !---------------------------! <78FD/EH>
  K ! Dim'd Variable Table      !
    !---------------------------! <78FB/CH>
  A ! Simple Variable Table     !
  R !---------------------------! <78F9/AH>
  E ! Program Statement Table   !
  A ===========================! <78A4/5H>
    ! Communication  Area       !
    !---------------------------! 7800H
    ! Video RAM                 !
    !---------------------------! 7000H
    !
```

All of these subdivisions of RAM used by Basic are not fixed and move up and down depending upon what actions are performed. Inserting or deleting a line from a Basic program alters the size of the PST. Similarly, defining a new variable, increases the length of the VLT. The String Area can only be changed by the CLEAR statement and this causes the relocation of the Stack. This is a very drastic action and results in a major "reset" of the various tables. How does the Interpreter keep track of these tables?

Since the origin of the tables may shift, their addresses are kept as POINTERS in the Communications Area of the VZ. I trust that there is beginning, some dawning of the relevance of information given in the VZ Technical Manuals.

On page 11 or 20, of the VZ 200 or 300 manual respectively, the various pointers are given. Olney page 125 and 129-130 provides a clear description of the foregoing. If you have these publications, then spend some time understanding the structure of a Basic program in memory.

The manner in which information is packed in each table is a very interesting topic, but will be left for another time as it is not required at present.

FAST BASIC PART I CONTINUED

I. RESERVING RAM FOR MACHINE CODE.

I WILL DISCUSS THE VARIOUS WAYS THAT MACHINE CODE (M/C) CAN BE LOCATED IN MEMORY ALONG WITH THE BASIC PROGRAM. THEY FALL INTO 4 TYPES-

I. LOADED INTO RAM WITH THE BASIC PROGRAM - APPENDED OR EMBEDDED.
II. POKED INTO RAM FROM THE BASIC PROGRAM - SET-UP TIME REQUIRED.
III. ALREADY EXISTING IN ROM - READY TO RUN.
IV. LOADED INTO RAM AS SEPERATE PROGRAMS - TWO OR MORE MODULES.

TYPE I. CAN BE LOCATED BELOW THE PST AND ABOVE THE COMMUNICATION AREA. THE COPYPRO PROGRAM OF LARRY TAYLOR USES THIS TECHNIQUE. METHOD 1 IN THE VZ TECHNICAL MANUALS DESCRIBE THE METHOD. METHOD 2 IN THE TECHNICAL MANUALS DESCRIBES HOW TO LOCATE M/C ABOVE THE PST AND BELOW THE VLT. OLNEY ON PAGE 47 ALSO DESCRIBES THIS TECHNIQUE. THE LATTER TECHNIQUE IS A BETTER METHOD THAN LOADING M/C BELOW THE PST. BOTH METHODS HAVE THE ADVANTAGE THAT A SINGLE MODULE, OF COMBINED M/C AND BASIC, IS LOADED INTO CO-JOINED MEMORY LOCATIONS.

TYPE II. METHODS HAVE A NUMBER OF VARIANTS. GENERALLY THE M/C IS HELD IN DATA STATEMENTS WITHIN THE BASIC PROGRAM LOCATED IN THE PST. THE THING THAT IS INTERESTING ABOUT THIS TECHNIQUE IS THAT THE M/C CAN BE POKED INTO THE BASIC PROGRAM AND EMBEDDED IN THE PST, POKED INTO FSL OR RESERVED MEMORY, OR PUT INTO THE VLT OR STRING AREA. IT IS OBVIOUSLY VERY FLEXIBLE.

TECHNIQUES THAT MODIFY THE PST IMPLY THAT THE POKEING OF THE M/C NEED ONLY BE DONE ONCE AND THAT IT IS SUBSEQUENTLY LOADED ALONG WITH THE BASIC PROGRAM. THE OTHER TECHNIQUES ARE NOT PARTICULARLY MEMORY EFFICIENT, AS TWO COPIES OF THE M/C ARE HELD IN THE PST, IN THE DATA STATEMENTS, AND ALSO IN ITS RUN-TIME LOCATION. METHOD 3 IN THE VZ TECHNICAL MANUALS LOWERS THE TOP-OF-MEMORY TO CREATE A RESERVED ARE OF RAM BELOW THE DOS VECTOR AND ABOVE THE STRING AREA.

OLNEY ALSO DESCRIBES THE METHOD ON PAGES 37-38. FOLLOWING THIS LOWERING IT IS NECESSARY TO RESET ALL OF THE BASIC POINTERS WITH A CLEAR NOW YOU UNDERSTAND WHY!!. THIS IS BY FAR AND WIDE THE BEST METHOD TO LOCATE M/C AS IT GIVES IT AN UNAMBIGUOUS AND PROTECTED AREA IN WHICH TO OPERATE. IT IS EASY TO DEBUG ALSO AS THE AREA CAN BE DISASSEMBLED IF REQUIRED. ONCE A RESERVED AREA OF MEMORY IS MADE, THE M/C CAN BE POKED IN AS IS DONE IN MY TONEGEN PROGRAM, OR IT MAY BE LOADED IN FROM DISK AS IS DONE WITH MOVEUP IN MY LIVENUP PROGRAM. THE MEMORY MAP FOR THIS ARRANGEMENT IS AS FOLLOWS -

```
 ============================!  <IY REG>
  TOP OF PHYSICAL MEMORY     !
  DOS VECTOR ON DISK SYSTEMS !
 ============================!
                             !
  RESERVED TOM AREA          !
                             !  <788E/FH>
 ============================!  <78B1/2H>
  STRING AREA                !
 ----------------------------   <78A0/1H>
  STACK                      !
 ----------------------------!  <78E8/9H>
                             !
! FREE SPACE LIST            !
!                            !
!                            !
```

CONTINUED NEXT ISSUE . . .

RENEWING PRINTER RIBBONS

BY JOHN LUXTON

WHO HAS A PRINTER THAT USES A HALF INCH OR 12MM PLASTIC BASE CARBON TAPE? WHO WOULD LIKE TO BE ABLE TO RELOAD THE CASSETTE AND SAVE A BUNDLE? WELL, IF YOUR PRINTER USES A CASSETTE WHICH EXPOSES ABOUT 26CM OF TAPE WHICH THE PRINT-HEAD WORKS ON, OR MAYBE OTHER CONFIGURATIONS, I MAY BE ABLE TO HELP.

MY PRINTER IS A COMPUTE MATE 130, MODEL CPB80. ACCORDING TO THE BOX OF A PELIKAN RIBBON I ONCE BOUGHT, THIS CARTRIDGE IS USED BY QUITE A NUMBER OF PRINTERS INCLUDING A COMMODORE 4023 P, A SEKONIC, SHINWA AND PROBABLY OTHERS. A GLANCE AT A CATALOGUE REVEALS MANY OTHER SIMILAR TYPES. ALSO THE CARTRIDGE I USE HAS A PROJECTING KNOB AT THE PINCH ROLLER END FOR MANUAL REWINDING. THE OPPOSITE SIDE HAS THE SOCKET WHICH IS ENGAGED BY THE PRINTER MECHANISM FOR ADVANCING THE RIBBON.

LAST YEAR WHEN WE WERE IN SYDNEY I GOT A YELLOW PAGES TO SEE IF I COULD FIND A SUPPLIER OF BULK RIBBON TO REFILL SPENT CARTRIDGES. AFTER A FEW DISCOURAGING CALLS I DISCOVERED A FIRM, LAZARUS RIBBONS, OF 70 WOLSELEY RD., MOSMAN, 2088, PHONE (02) 960 2737, WHO WOULD SELL ME A 1000 FOOT ROLL OF RIBBON FOR A REASONABLE PRICE. THEY WERE AWAITING STOCK FROM THE U.S.A., SO IN DUE COURSE I RECEIVED THE ROLL OF RIBBON POSTED TO MY HOME FOR $29.73 ALL UP!

I FIRST EXPERIMENTED BY REMOVING THE TOP OF THE CARTRIDGE BY THE JUDICIOUS USE OF A KNIFE. THESE TOPS ARE HELD BY DOWELS, OR OCCASIONALLY WITH SMALL SCREWS AND DOWELS, BUT CAN BE REMOVED WITH CARE. HOWEVER THIS METHOD IS NOT RECOMMENDED, AS SPRINGS AND THINGS CAN CAUSE PROBLEMS. THE BETTER WAY TO GO ABOUT THE JOB IS TO PULL OUT THE RIBBON FROM THE DELIVERY END UNTIL THE CARTRIDGE IS EMPTIED.

BEFORE PROCEEDING A FEW THINGS ARE REQUIRED TO DO THE JOB. A VIDEO/AUDIO TAPE SPLICING JIG, TANDY CAT. NO. 44-9570 AT ABOUT $6.95 IS HANDY. I PREFER THE WIDER 3M MAGIC MENDING TAPE, BUT HALF INCH VIDEO SPLICING TAPE IS O.K. A SHARPER RAZOR BLADE THAN SUPPLIED BY TANDY IS ALSO USEFUL. WINDING THE NEW RIBBON INTO THE CARTRIDGE CAN BE DONE MANUALLY, BUT TAKES TIME. I USE MY NEWLY ACQUIRED MAKITA CORDLESS DRILL COME SCREWDRIVER, SET IN REVERSE. IT IS SLOW ENOUGH, BUT DOES THE JOB VERY EFFICIENTLY.

WHEN THE TAPE IS ALL EMPTIED FROM THE CARTRIDGE, CUT THE TAPE ACROSS THE FRONT, AND SPLICE ON THE END OF THE REPLACEMENT TAPE. MAKE A GOOD SPLICE AND TRIM ANY EXCESS STICKY TAPE, THEN PULL THE NEW RIBBON THROUGH THE CARTRIDGE FROM THE DELIVERY END. IF YOUR SPLICE WAS FAULTY AND BREAKS INSIDE, YOU WILL HAVE TO TAKE THE LID OFF AND FEED THE TAPE THROUGH, MAKING SURE YOU DON'T END UP WITH SPRINGS AND PINCH ROLLERS EVERYWHERE.

PULL OUT A FOOT OR SO OF THE NEW RIBBON, THEN DEVISE A WAY OF MEASURING OFF THE REQUIRED AMOUNT FROM THE ROLL. USE CLEAN BOXES OR SUCH TO HOLD THE LOOSE RIBBON AND KEEP IT CLEAN. I HAVE FOUND THAT THE CARTRIDGES I USE HOLD FROM 25 TO 30 METRES. IT SEEMS TO VARY A BIT. YOU CAN BEST WORK OUT YOUR OWN METHOD OF MEASURING THE REQUIRED LENGTH.

NOW TO LOAD THE NEW RIBBON INTO THE CARTRIDGE. THE WINDER KNOB ON MY CARTRIDGES GOES ANTI-CLOCKWISE, SO I HAVE TO SET REVERSE ON THE DRILL, FASTEN THE CHUCK TO THE WINDING KNOB WITHOUT USING UNDUE FORCE. THEN I HOLD THE CARTRIDGE VERTICALLY WITH THE DELIVERY END BETWEEN MY KNEES, AND HOLDING THE DRILL IN MY RIGHT HAND COMMENCE LOADING, GUIDING THE TAPE THROUGH MY LEFT HAND TO AVOID ANY KINKING.

RENEWING RIBBONS CONTINUED

WHEN A COUPLE OF FEET OF RIBBON ARE LEFT, UNCHUCK THE DRILL. THEN CAREFULLY SPLICE EACH END, TAKING CARE NOT TO TWIST THE RIBBON, AND PUTTING THE SPLICING TAPE ON THE SHINY SIDE. IF USING THE WIDER TAPE, CAREFULLY TRIM WITH SHARP SCISSORS ANY STICKY TAPE EITHER SIDE OF THE RIBBON, THEN WIND THE EXCESS INTO THE CARTRIDGE. I SOMETIMES AGAIN CHUCK UP THE DRILL AND RUN THE RIBBON THROUGH TO CHECK ON THE SPLICE, BUT IF QUALITY SPLICING TAPE IS USED, THERE SHOULD BE NO WORRY. AT THE MOMENT I HAVE RELOADED 4 CARTRIDGES, AND FIND THAT I CAN DO ONE UNDER 20 MINUTES.

THE SAVINGS ARE OBVIOUS. AT LEAST 10 CARTRIDGES CAN BE RELOADED FROM A 1000 FOOT ROLL OF RIBBON. DEPENDING WHERE I BUY RIBBON CARTRIDGES, AND THE BRAND, THE COST CAN AVERAGE,SAY, $20. THAT'S AN OUTLAY OF AROUND $200 AGAINST THE $30 I PAID FOR THE ROLL OF RIBBON. SO FAR I CAN'T VOUCH FOR THE QUALITY OF THE RIBBON, BUT WAS ASSURED IT WAS BEST QUALITY FROM THE U.S.A.

I FOUND IT ADVISABLE TO MAKE UP A COUPLE OF CHEEKS THE SIZE OF THE ROLL FROM MASONITE. THE HOLE IN THE CENTRE OF THE ROLL WAS ONE AND A HALF INCHES, SO I CUT A CIRCLE FROM HALF INCH PINE BOARD WITH A HOLE CUTTER, AND USED THE PIECE FOR THE CENTRE OF THE ROLL. FORTUNATELY I HAVE A LATHE TO FINISH AND SAND THE CHEEKS AND CENTRE TO SIZE. POSSIBLY THE RIBBON CAN BE SPOOLED OFF WITHOUT SPILLING, BUT I TOOK NO CHANCES. A QUARTER INCH MACHINE SCREW AND NUT FINISHED OFF THE JOB.

AS I INDICATED, I CAN ONLY SPEAK FOR THE CARTRIDGE USED IN MY PRINTER, BUT I SHOULD IMAGINE THAT MOST TYPES OF CARTRIDGE CONTAINING PLASTIC HALF INCH RIBBON CAN BE DEALT WITH THIS WAY. COULD BE AN OPENING FOR A COTTAGE INDUSTRY!.

DOUBLE PRECISION ARITHMETIC
BY NEVILLE HUGHES

WELL I HAVE FOUND A VZ 200/300 BASIC ROM FUNCTION THAT IS NOT MENTIONED IN ANY OF MY LITERATURE ON COMPUTERS. THE HASH SYMBOL (#) AFTER THE LAST NUMBER IN A MATHEMATICAL SUM GIVES YOU 'DOUBLE PRECISION' IN THE ANSWER UP TO 14 FIGURES IN SAID ANSWER. TRY THE FOLLOWING:

```
2222 * 2222         <RETURN> - ANSWER IS 4.93728E+06
2222 * 2222#        <RETURN> - ANSWER IS 4937284
2222222 * 2222222   <RETURN> - ANSWER IS 4.93827E12
2222222 * 2222222# <RETURN> - ANSWER IS 4938270617284
```

MUCH MORE ACCURATE WHEN YOU HAVE MORE THAN SIXTEEN FIGURES IN THE ANSWER YOU GET A D+OR-05 OR WHATEVER. THE 'D' STANDS FOR DOUBLE PRECISION. HAVE FUN WITH THIS, I SURE HAVE. NEVILLE HUGHES.

NEVILLE IS RIGHT UP TO A POINT, THE '#' SYMBOL WAS NOT IN GENERAL LITERATURE THAT CAME WITH THE VZ. COLIN BRIDGE, A LOCAL MEMBER REMINDED ME THAT THERE WAS AN ADDENDUM THAT CAME WITH SOME VZ 300'S ON HOW TO ACHIEVE DOUBLE PRECISION ARITHMETHIC WHICH IS REPRODUCED BELOW.

DOUBLE PRECISION ARITH. ADDENDUM

THIS IS A RECOMMENDED METHOD FOR THOSE USERS WHO WANT TO USE THE COMPUTER TO DO DOUBLE PRECISION ARITHMETHICS. THE SIMPLE EXAMPLE BELOW SHOWS THE IMPLEMENTATION.

```
    10 CLEAR 100
    20 A$=STR$(10/9#)
    30 PRINT"A  =";A$
    40 B$=STR$(VAL(A$)*VAL(A$))
    50 PRINT"A * A";B$
```

IN LINE 10, THE PROGRAM RESERVES 100 BYTES FOR STRING OPERATION. IN LINE 20, THE VARIABLE IS DEFINED AS A STRING WITH AN EQUIVELENT VALUE OF 10/9. THE '#' SIGN FOLLOWING 10/9 TELLS THE COMPUTER THAT THE VALUE 10/9 SHOULD BE IN DOUBLE PRECISION REPRESENTATION.

LINE 30 PRINTS OUT THE VALUE OF 10/9 IN DOUBLE PRECISION VALUE. IN LINES 40 AND 50, THE VALUE OF 10/9*10/9 IS CALCULATED AND PRINTED OUT IN DOUBLE PRECISION VALUE.

TYPE "RUN" AND THE COMPUTER WILL HAVE THE RESULT AS FOLLOWING.

```
RUN
A  = 1.1111111111111
A*A= 1.234567901234568
```

DOUBLE PRECISION ARITH. IN USE

ALTHOUGH THE ABOVE WORKS AS DESCRIBED, THERE IS AN EASIER WAY SIMILAR TO WHAT NEVILLE WAS USING AND YOU DON'T HAVE TO USE THE STR$ AND VAL FUNCTIONS. IN THE MEANTIME TYPE IN THE DEMONSTRATION PROGRAM BELOW AND SEE FOR YOURSELF.

NOTE: USE POKES TO DEFINE VARIABLES IN LINES 20 TO 40 AS SHOWN IN REM STATEMENTS IF YOU HAVEN'T AN EXTENDED BASIC.

```
10 CLS:PRINT
20 DEFINT A:A=1/3:PRINT A;TAB(20)"INT":REM ... POKE 30977,2
30 DEFSNG B:B=1/3:PRINT B;TAB(20)"SNG":REM ... POKE 30978,4
40 DEFDBL C:C=1/3:PRINT C;TAB(20)"DBL":REM ... POKE 30979,8
50 C0=1/3#:PRINT C0;TAB(20)"DBL"
60 BZ=1#/3:PRINT BZ;TAB(20)"SNG"

RUN  <RETURN>

0                   INT; LINE 20
.333333             SNG; LINE 30
.3333333432674408   DBL; LINE 40
.3333333333333333   DBL; LINE 50
.333333             SNG; LINE 60
```

LINE 20 DEFINES A AS AN INTEGER VARIABLE AND PRINTS RESULT.

LINE 30 DEFINES B AS A SINGLE PRECISION VARIABLE AND PRINTS RESULT.

LINE 40 DEFINES C AS A DOUBLE PRECISION VARIABLE AND PRINTS RESULT.

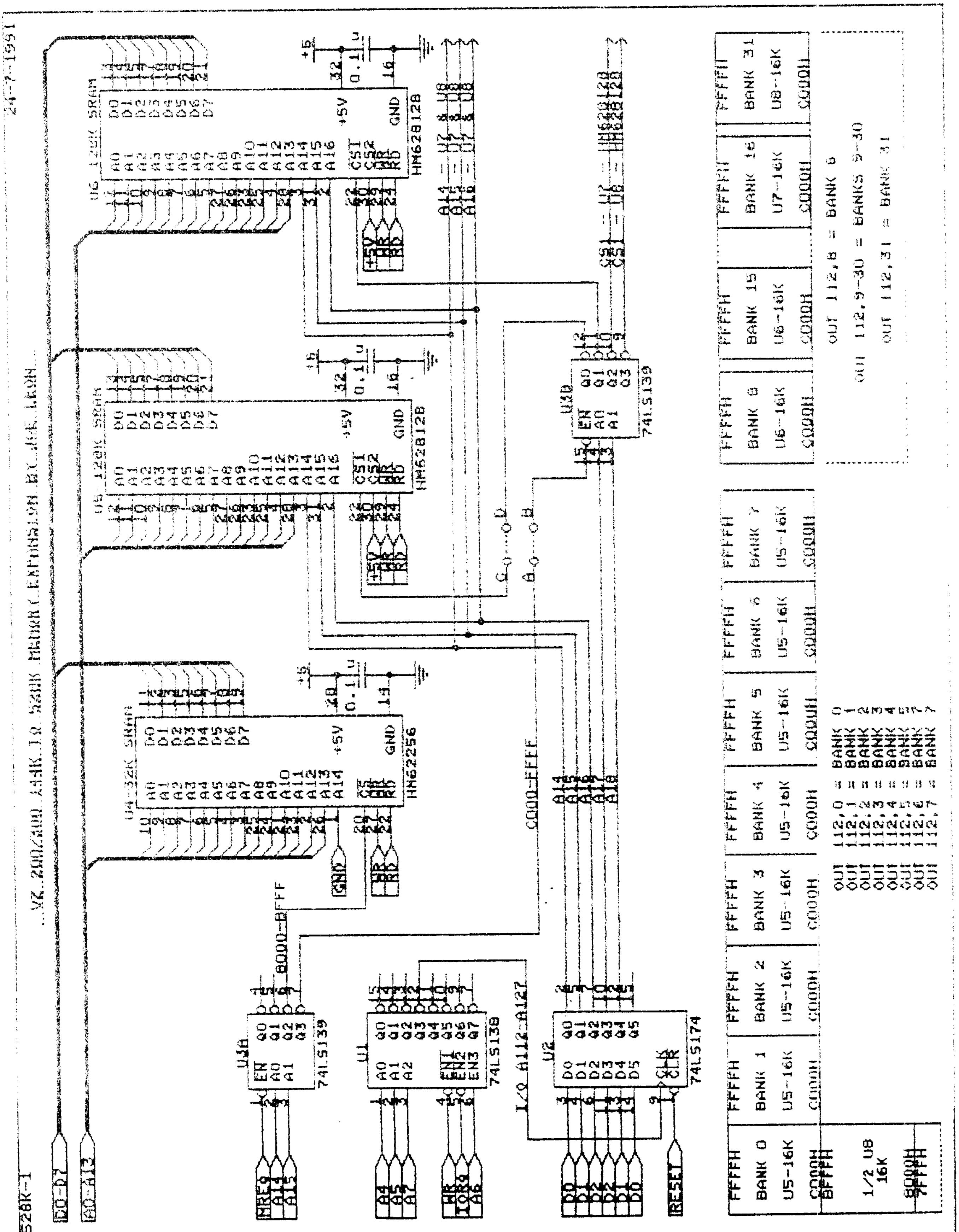
LINE 50: THE '#' SYMBOL IS USED FOR INCREASED ACCURACY.

LINE 60: ALTHOUGH THE '#' SYMBOL WAS USED IN THE FORMULA THE RESULT IS SINGLE PRECISION BECAUSE LINE 30 DEFINED ANY VARIABLE STARTING WITH 'B' AS SINGLE PRECISION.

NOTE: ONLY A DOUBLE PRECISION VARIABLE CAN HOLD A DOUELE PRECISION SUM.

VZ 200/300 144K-528K MEMORY 36/16

EXPANSION BY JOE LEON



530K: THAT'S A LOT OF MEMORY FOR THE HUMBLE VZ AND WOULD SHAME MANY A COMPUTER. THE MEMORY IS MADE UP IN THE FOLLOWING WAY:

```
    7800H-7FFFH (30720D-32767D) - START OF USER MEMORY.
    8000H-BFFFH (32768D-49151D) - U4-32K SRAM (ONLY 16K USED).
    C000H-FFFFH (49152D-65535D) - U5-U8-128K SRAM/S.
```

MOST WOULD ONLY ENABLE 144K (32K+128K) OF MEMORY LIKE I DID AND IS ADDITIONAL MEMORY THEY WOULD HARDLY EVER USE. AND NOW TO THE CIRCUIT.

U1-74LS138: THIS 3 OF 8 DECODER IC IS THE SAME AS USED IN THE 64K RAM PACK AND DECODES I/O ADDRESS 112-127 AND IS USED TO INITIATE BANK SWITCHING.

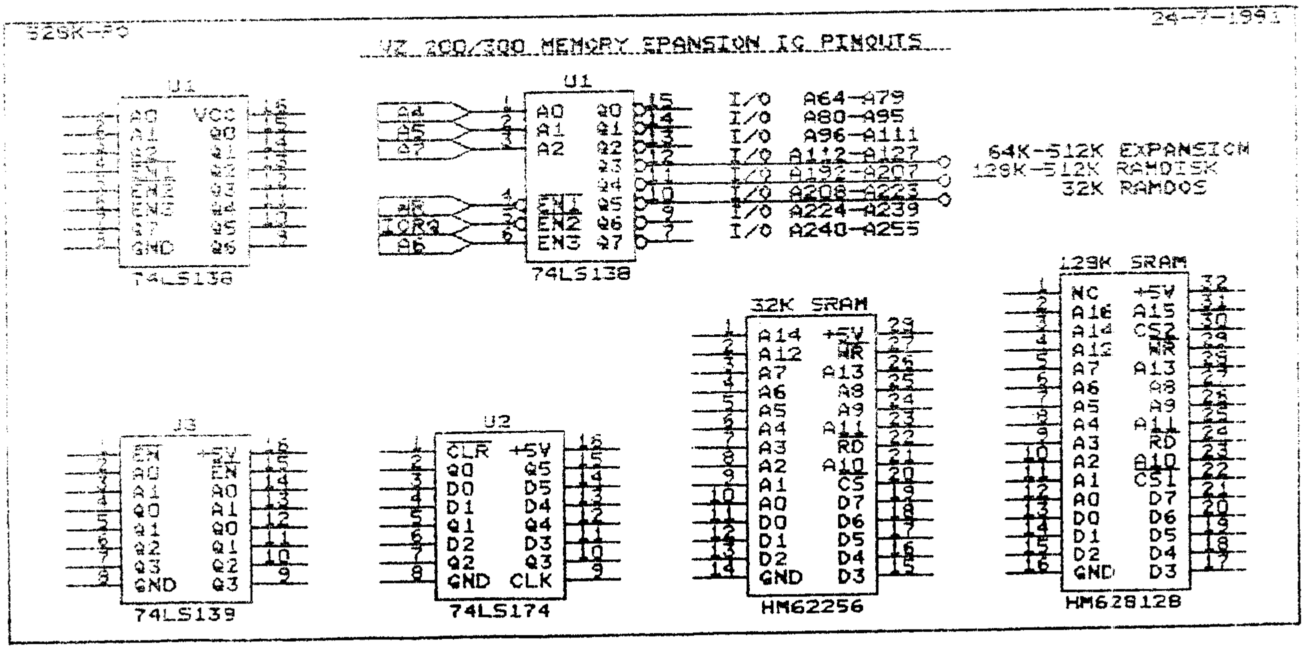
U2-74LS174: THIS 6 BIT LATCH IS ACTIVATED BY U1 AND GENERATES THE EXTRA ADDRESS LINES NEEDED FOR BANK SWITCHING. O/P'S Q0-Q2 GENERATE A14-A16 AND IS USED TO BANK SWITCH U5-U8. AT POWER UP OR RESET ALL BITS ARE LO SO BANK 0 IN U5 IS SELECTED.

U3A-1/2 74LS139: THIS 1 OF 3 DECODER IC IS DESIGNED TO DECODE IN 16K BLOCKS AND THE FIRST 2 O/P'S AREN'T USED. Q2 ENABLES U4 VIA ITS CS BAR (CHIP SELECT) IN THE ADDRESS RANGE 8000H-BFFFH. Q3 SELECTS THE BANK SWITCHING ADDRESS RANGE OF C000H-FFFFH.

U3B-1/2 74LS139: THE 2ND HALF OF THIS IC IS USED TO SELECT 1 OF 4 (U5-U8) 128K SRAMS AND ONLY ONE CAN BE SELECTED AT ANY ONE TIME. U2-Q3 & Q4 (A17 & A18) CONTROL WHICH 128K SRAM IS SELECTED. WITH ALL BITS LO ON U2, Q0 ON U3B GOES LO AND ENABLES U5 VIA ITS CS.

U3B-OPTIONAL: IF YOU WANT TO USE ONLY ONE 128K SRAM THEN YOU CAN BYPASS U3B. CUT LINK BETWEEN C & D AND CONNECT A TO C. LEAVE ON LINK BETWEEN A & B THOUGH AS IT WILL HAVE NO EFFECT ON CIRCUIT AND WILL SAVE FROM CONNECTING B TO +5V. IF YOU WANT TO ADD MORE 128K SRAMS CHIPS LATER THEN CHANGE LINKS BACK TO ORIGNAL CIRCUIT.

74LSXX IC'S: ALTHOUGH 74LSXX CHIPS ARE SHOWN, USE 74HCT138, 74HCT139 & 74HCT174 INSTEAD IF YOU CAN GET THEM AS THEY HAVE A LOWER POWER CONSUMPTION, BUT THEY ARE A LITTLE BIT DEARER.



144K-528K MEMORY EXP. CONT.

CONSTRUCTION GUIDE: AT THIS TIME I CAN ONLY GIVE A ROUGH GUIDE. I USED DAVID NECOMBES'S 32K RAM BOARD (SEE ISSUE # 20, PAGE 8) TO CONSTRUCT MY PROTOTYPE. I LEFT 2 X 8K SRAMS (IC 1 & 2) OCCUPYING 8000H-BFFFH MEMORY LOCATIONS. U1-U3 WERE PLACED ON A PIECE OF VEROBOARD. U4 WAS NOT REQUIRED.

IN IC 4'S PLACE I INSTALLED A 32 PIN (CUT DOWN FROM 40) IC SOCKET WITH PINS 1, 2, 3, 28, 31 AND 32 BENT OUT AT RIGHT ANGLES. THAT WAY THE REQUIRED CONNECTIONS WERE MADE TO THE SOCKET PINS AND NOT TO THE PINS OF THE 128K SRAM WHICH IS A BIT ON THE EXPENSIVE SIDE.

VZ 200 SUPER II BOARD: A SIMILIAR PROCEDURE COULD BE USED WITH THIS BOARD AS WELL. MY APOLOGIES FOR NOT GIVING MORE CONSTRUCTION DETAILS AT THIS TIME AS I DIDN'T HAVE TIME TO PREPARE MORE COMPREHENSIVE DETAILS.

144K-512K PCB: THE INTENTION IS TO EVENTUALLY PRODUCE A PCB FOR ABOVE WHICH WILL HAVE PROVISION FOR 1 X 32K AND 2 X 128K SRAMS (272K TOTAL). IT WILL ALSO ACCEPT A PIGGY BACK PCB WHICH WILL CONTAIN MY RAM DISK CONTAINING 32K SRAM AT 6000H-67FFH-RAM DOS AND 128K-256K AT 8000H-BFFFK-RAM DISK (288K TOTAL). ADDING BOTH TOGETHER THE GRAND TOTAL WOULD BE 560K OF MEMORY IN ONE PLUG IN CARTRIDGE FOR THE VZ.

BANK ARRANGEMENT: THE 64K RAM EXPANSION HAS BANK 0 FIXED AT ADDRESS 8000H-BFFFH AND USES COMPLEX DECODING TO ACHIEVE IT. ALSO SELECTING BANK 0 WILL ACTUALLY GET YOU BANK 1.

MY 144K-512K DESIGN HAS ALL 8-32 BANKS OCCUPYING ADDRESS RANGE 8000H-BFFFH. ALL PROGRAMS WRITTEN FOR THE 64K RAM PACK WILL WORK IN MY DESIGN BUT YOU WILL BE ABLE TO ACCESS BANKS 1-3 ONLY. MOST PROGRAMS WILL BE FAIRLY EASY TO MODIFY FOR FULL USE AS IT'S JUST A MATTER OF ADDING EXTRA OUT 112,X COMMAND/S TO SELECT DESIRED BANK/S.

128K SRAM: ONLY [illegible] OF U4 IS USED AND THE OTHER [illegible] IS DISABLED BY CONNECTING PIN 1-A14 LO. DOING IT THIS WAY MADE FOR A SIMPLER CIRCUIT USING LESS DECODING IC'S.

IC PINOUTS: TO MAKE CONSTRUCTION EASIER PINOUTS OF ALL IC'S USED ARE SHOWN. ALSO SHOWN ARE ALL THE DECODED I/O ADDRESSES OF U3 SOME OF WHICH WILL BE USED IN A FUTURE PROJECT.

PURCHASING 32K & 128K MEMORY IC'S:
THE ABOVE IC'S CAN BE PURCHASED FROM A FEW SUPPLIERS. THE CHEAPEST I FOUND SO FAR IS AVAILABLE FROM:

R-A-E INDUSTRIAL ELECTRONICS PTY LTD
62 MOORE STREET AUSTINEMER NSW 2515
PHONE: (022) 232 6933 FAX: (042) 68 1075

1 OFF HM628128LP-10 128K X 8 CMOS SRAM - $27.83+20%=$33.40
1 OFF HM62256LP-12 32K X 8 CMOS SRAM - $5.70+20%=$ 6.84
POST & PACKING: REGISTERED MAIL: $3.50 OVERNIGHT COURIER: $5.00

NOTE: THE PRICES QUOTED ARE FOR HM-LP (HITACHI MEMORY-LOW POWER) VERSIONS WHICH USE LESS CURRENT AND CAN BE BATTERY BACKED. PRICES AND AVAILABILITY COULD CHANGE SO CHECK FIRST BEFORE ORDERING.

THE TWO MEMORY CHIPS WILL SET YOU BACK ABOUT $45.00 WHICH IS VERY CHEAP FOR 144K OF MEMORY FOR THE VZ. THAT'S CHEAPER THAN YOU CAN BUY THE VZ ITSELF NOW.

.. CONTINUED NEXT ISSUE ..

*** DAVE MITCHELL SOFTWARE ***

E & F WP PATCH 3.3: PATCH 3.3 WRITTEN BY DAVE MITCHELL WILL CONVERT YOUR E & F TAPE WORD PROCESSOR FOR FULL DISK USE WHILE RETAINING ALL ORIGINAL FUNCTIONS. IT ALSO HAS SHIFT LOCK AND PRINTER CONTROL CODES WHICH CAN BE EMBEDDED IN TEXT AND SAVED TO TAPE OR DISK.

BASTWP.FP: THIS UTILITY PROVIDED WITH PATCH 3.3 WILL CONVERT BASIC PROGRAMS AND EDASM. SOURCE CODE FILES INTO WORD PROCESSOR FILES.

PRICE: AUS/NZ AUS20.00 - UPDATE - AUS-$10.00 - NZ-AUS11.00.

EXTENDED DOS V1.3: THESE COMMANDS ARE AT YOUR DISPOSAL: MERGE, READ, TYPE, LIBB, LIBRB, OLD, OLD., DEC, HEX, MENU, CODE, LTAB, MOVE AND UPDATE STATUS, AND LSTATUSA. STATUSA AND LSTATUSA ALSO WORKS WITH VERSION 1.2 DOS

PRICE: AUS15.00 - POSTAGE INCLUDED

MENU DIR/FILE COPIER: THIS UTILITY WILL READ YOUR DISK DIRECTORY AND PRESENT IT WITH SEVERAL OPTIONS. USING THE CURSOR YOU CAN RUN/BRUN ANY PROGRAM OR SELECT FILE COPY, REN, ERASE, DRIVE 1 OR 2, ETC. BESIDES COPYING TEXT AND BINARY FILES ALL OTHER FILES CAN BE COPIED AS WELL EXEPT FOR DATA FILES

PRICE: AUS15.00 - POSTAGE INCLUDED

FOR PURCHASE OR INFORMATION CONTACT:

DAVE MITCHELL 24 ELPHINSTONE STREET
NORTH ROCKHAMPTON QUEENSLAND 4701
AUSTRALIA - PHONE: (079) 27 8519

*** PETER HICKMAN SOFTWARE ***

THE [illegible] DISASSEMBLER: WHAT, ANOTHER DISASSEMBLER? [illegible] THIS ONE IS DIFFERENT! THIS PROGRAM IS ENTIRELY WRITTEN IN MACHINE CODE. IT ACTUALLY RUNS ABOUT 40 TIMES FASTER THAN D.S.E.'S DISASSEMBLER (OR ANY ONE ELSE'S). IT WILL DISASSEMBLE ANY PROGRAM THAT YOU CAN BLOAD INTO MEMORY. IT WORKS WITH ANY [illegible] CONFIGURATION. IT DISASSEMBLES EVEN THE 58 EXTRA Z80 OPCODES THAT ZILOG DOESN'T ADMIT TO.

PRICE: AUS25.00 - PRICE INCLUDES HARDCOPY MANUAL.
TAPE AND DISK VERIONS AVAILABLE.

[illegible] MODEM SOFTWARE: DID YOU WANT TO TALK TO OTHER COMPUTERS VIA A MODEM? DID YOU BUY THE DSE TERMINAL EPROM, ONLY TO DISCOVER THAT IT ONLY WORKS WITH TAPE. IT ONLY ALLOWS YOU TO PRINT FILES, NOT SAVE THEM OR SEND THEM!

YOUR PROBLEMS ARE SOLVED! THE HICKMAN BROTHERS, PETER AND ANDREW, HAVE A BRAND NEW PROJECT WHICH WILL ALLOW YOU TO SEND, RECEIVE & SAVE FILES VIA A MODEM. IT WORKS WITH DISK!

SALE PRICE: $25.00 - INCLUDED ARE INSTRUCTIONS FOR THE HARDWARE MODIFICATIONS. A SMALL MODIFICATION IS NEEDED TO YOUR DISK CONTROLLER. YOUR USER GROUP MAY HELP YOU MODIFY YOUR COMPUTER TO USE THIS EXCITING NEW SOFTWARE

THE MANUAL IS SUPPLIED ON DISK FOR PRINTING OUT WITH YOUR DISK VERSION OF E & F W/PROCESSOR. IF YOU DO NOT OWN AN E & F W/PROCESSOR THEN PLEASE ENCLOSE ANOTHER $5.00 (TOTAL $30.00) FOR PHOTOCOPYING AND POSTAGE OF THE MANUAL.

FOR PURCHASE OR INFORMATION CONTACT:
PETER HICKMAN PO BOX 8 WERRINGTON 2747

* * CONTRIBUTIONS TO THE JOURNAL * *

IF YOU ARE THINKING OF CONTRIBUTING TO THE JOURNAL THE PREFERED FORMAT IS BASIC LISTINGS, WORD PROCESSOR OR SOURCE CODE FILES ON TAPE OR DISK. FILES FROM THE FOLLOWING WORD PROCESSORS CAN BE ACCEPTED :-

[illegible] WORDPRO DISK PATCH 3.1-3.3, WORDPRO CARTRIDGE, WORDPRO PATCH, MOST SOURCE CODE FILES AND ALL QUICKWRITE WORD PROCESSOR FILES.

* * CLUB MEETINGS - ALL WELCOME * *

FIRST FRIDAY OF MONTH

* * FUTURE MEETINGS - NEW VENUE * *

FROM JULY 5 ON WE WILL MEET AT JOE LEON'S PLACE OR OTHER MEMBERS PLACES AS DECIDED BY MEMBERS.

AUGUST 2 CLUB MEETING AT

JOE LEON 20 DRURY STREET WALLSEND 2287 (049) 512 756

SEPT 6 - OCT 4 - NOV 1 - DEC 6

[illegible] SUTTON OR LOCAL CLUB MEMBERS FOR SEPTEMBER [illegible] BUT AT THIS TIME IT HASN'T BEEN FINALISED.

MEETINGS WILL BE ONCE A MONTH AS BEFORE WITH THE DATES BEING [illegible] OF THE MONTH. BECAUSE OF SOME LOCAL MEMBERS HAVING TO WORK SHIFTWORK, MEETING DATES WILL BE ADJUSTED TO ACCOMODATE THEM. PLEASE CHECK WITH JOE LEON FIRST BEFORE COMING OUT.

* * [illegible] COMMITTEE & SUBSCRIPTIONS * *

[illegible]
COMMITTEE MEMBERS - GARY BULLEY - COLIN BRIDGE - PETER JONES

SUBSCRIPTION TO [illegible] - AUST. - 3 ISSUES $11.00 - 6 ISSUES $21.00
JOURNAL - N.Z. - 3 ISSUES $13.00 - 6 ISSUES $25.00

FOR MORE INFORMATION CONTACT JOE LEON (049) 512 756
HUNTER VALLEY VZ USERS GROUP - PO BOX 151 JESMOND 2299 AUSTRALIA

NOTE:- PRICES INCLUDE POST & PACKING

* * OTHER VZ USER GROUPS & PUBLICATIONS * *

VZ [illegible] - VZ MAGAZINE - 6 ISSUES - $18.00 PER ANUM
HARRY [illegible] 12 THOMAS SREET MITCHAM VICTORIA 3132

WAVZ - WESTERN AUSTRALIA VZ USER GROUP
GRAEME [illegible] PO BOX 388 MORLEY W.A. 6062

BRISBANE VZ USERS WORKSHOP - C/O 65 TINGALPA ST. WYNUM WEST 4178
SOFTWARE FOR SALE - DISK MENU

SAPPHIRE PRODUCTIONS - VZ DISK MAGAZINE - $20.00 PER ANUM
CONTACT JASON OAKLEY PO BOX 600 TAREE NSW 2430

NOTE:- WHEN WRITING TO ANY ABOVE OR H.V.VZ. USERS' GROUP FOR INFORMATION PLEASE ENCLOSE A S.S.A.E. OR NZ 2 INT. REPLY COUPONS.